Architecture of the IBM BG/Q

Argonne Training Program on Extreme Scale Computing

Scott Parker
Ray Loy
Argonne Leadership Computing Facility
8/03/2015

Argonne Blue Gene Timeline

- **1999**
  - IBM begins $100 million R&D project on Blue Gene architecture
  - Initial target was protein folding applications
  - Design evolved out of the Cyclops64 and QCDOC architectures
- **2004:**
  - Blue Gene/L introduced
  - LLNL 90-600 TF system #1 on Top 500 for 3.5 years
- **2005:**
  - Argonne accepts 1 rack (1024 nodes) of Blue Gene/L (5.6 TF)
- **2006:**
  - Argonne Leadership Computing Facility (ALCF) founded
  - ANL working with IBM on next generation Blue Gene
- **2008:**
  - ALCF accepts 40 racks (160k cores) of Blue Gene/P (557 TF)
- **2009:**
  - ALCF approved for 10 petaflop system to be delivered in 2012
  - ANL working with IBM on next generation Blue Gene
- **2012:**
  - 48 racks of Mira Blue Gene/Q (10 PF) hardware delivered to ALCF
  - Mira in production

Blue Gene in the Top 500

- Since being released 11 years ago, on the Top500 list:
  - A Blue Gene was #1 on half of the lists
  - On average 3 of the top 10 machines have been Blue Gene's
- Blue Gene/Q currently has 4 entries in top 10 of the Top500:
  - #3 – LLNL Sequoia, 96k nodes, 20PF
  - #5 – ANL Mira, 48k nodes, 10PF
  - #9 – Juelich Juqeen, 28k nodes, 5.8 PF
  - #10 - LLNL Vulcan, 24k nodes, 5 PF
- BG/P and BG/Q both held #1 on the Green500

Blue Gene DNA

- **Leadership computing power**
  - Leading architecture since introduction, #1 half Top500 lists over last 10 years
- **Low speed, low power**
  - Embedded PowerPC core with custom SIMD floating point extensions
  - Low frequency (L – 700 MHz, P – 850 MHz, Q – 1.6 GHz)
- **Massive parallelism:**
  - Multi/Many core (L - 2, P – 4, Q – 16)
  - Many aggregate cores (L – 208k, P – 288k, Q – 1.5M)
- **Fast communication network(s)**
  - Low latency, high bandwidth, torus network (L & P – 3D, Q – 5D)
- **Balance:**
  - Processor, network, and memory speeds are well balanced
- **Minimal system overhead**
  - Simple lightweight OS (CNK) minimizes noise
- **Standard Programming Models**
  - Fortran, C, C++, & Python languages supported
  - Provides MPI, OpenMP, and Pthreads parallel programming models
- **System on a Chip (SoC) & Custom designed Application Specific Integrated Circuit (ASIC)**
  - All node components on one chip, except for memory
  - Reduces system complexity and power, improves price / performance
- **High Reliability:**
  - Sophisticated RAS (reliability, availability, and serviceability)
- **Dense packaging**
  - 1024 nodes per rack

ALCF and the BG/Q Development

- **Over a three year period ANL collaborated with LLNL and IBM in joint research and development for the Blue Gene/Q providing input on design directions**
- **ANL and LLNL reviewed and provided feedback on several dozen technical milestone documents related to the design of the Blue Gene/Q:**
  - "BG/Q Design Trade-Off Study"
  - "BG/Q Core Choice Review"
  - "BG/Q Messaging Software Review"
  - "BG/Q Compute Node Kernel"
  - "API for Prefetcher"
  - …
- **Monthly conference calls to discuss BG/Q design aspects**
- **Quarterly on-site review meetings to review status and progress**
- **ANL & LLNL Statement-of-Work contracts specifying in detail the system specifications and deliverables**
- **Provided representative application benchmarks**
- **Provided IBM access to Intrepid and Mira for software development and testing at scale**

Evolution from P to Q

Design Parameters	BG/P	BG/Q	Difference
Cores / Node	4	16	4x
Hardware Threads	1	4	4x
Concurrency / Rack	4,096	65,536	16x
Clock Speed (GHz)	0.85	1.6	1.9x
Flop / Clock / Core	4	8	2x
Flop / Node (GF)	13.6	204.8	15x
RAM / core (GB)	0.5	1	2x
Mem. BW/Node (GB/sec)	13.6	42.6	3x
Latency (MPI zero-length, nearest-neighbor node)	2.6 µs	2.2 µs	~15% less
Bisection BW (32 racks)	1.39TB/s	13.1TB/s	9.42x
Network	3D Torus + Collectives	5D Torus	Smaller diameter
GFlops/Watt	0.77	2.10	3x
Instruction Set	32 bit PowerPC + DH	64 bit PowerPC + QPX	New vector instructions
Programming Models	MPI + OpenMP	MPI + OpenMP	
Cooling	Air	Water	

Mira Science Applications

BG/P version as-is on BG/Q

Apps	BQ/P Ratio	Comments
DNS3D	11.8	2048^3 grid, 16K cores, 64 ranks/node
FLASH	5.5 (9.1)	rtflame, 2K cores, 64 ranks/node rtflame, 16K cores, 8 ranks/node, 8 threads/rank, no MPI-IO
GFMC	10.5	c12-test, 2K cores, 8 ranks/node, 8 thrds/rank
GTC	10.8	M0720, 16K cores, 16 ranks/node, 4 thrds/rank
GFDL	11.9	Atm, 2K cores, 16 ranks/node, 4 thrds/rank
MILC	6.1	32^3x64 lattice, 2K cores, 64 ranks/node, no QPX
NEK	8.5	med case, 1K cores, 32 ranks/node, no QPX
NAMD	9.7	ATPase bmk, 2K cores, 16 ranks/node
GPAW	7.6	Au_bulk5x5x5, 2K cores, 16 ranks/node
LS3DF	8.1	ZnOTe, 8K cores, ESSLsmp, I/O sensitive

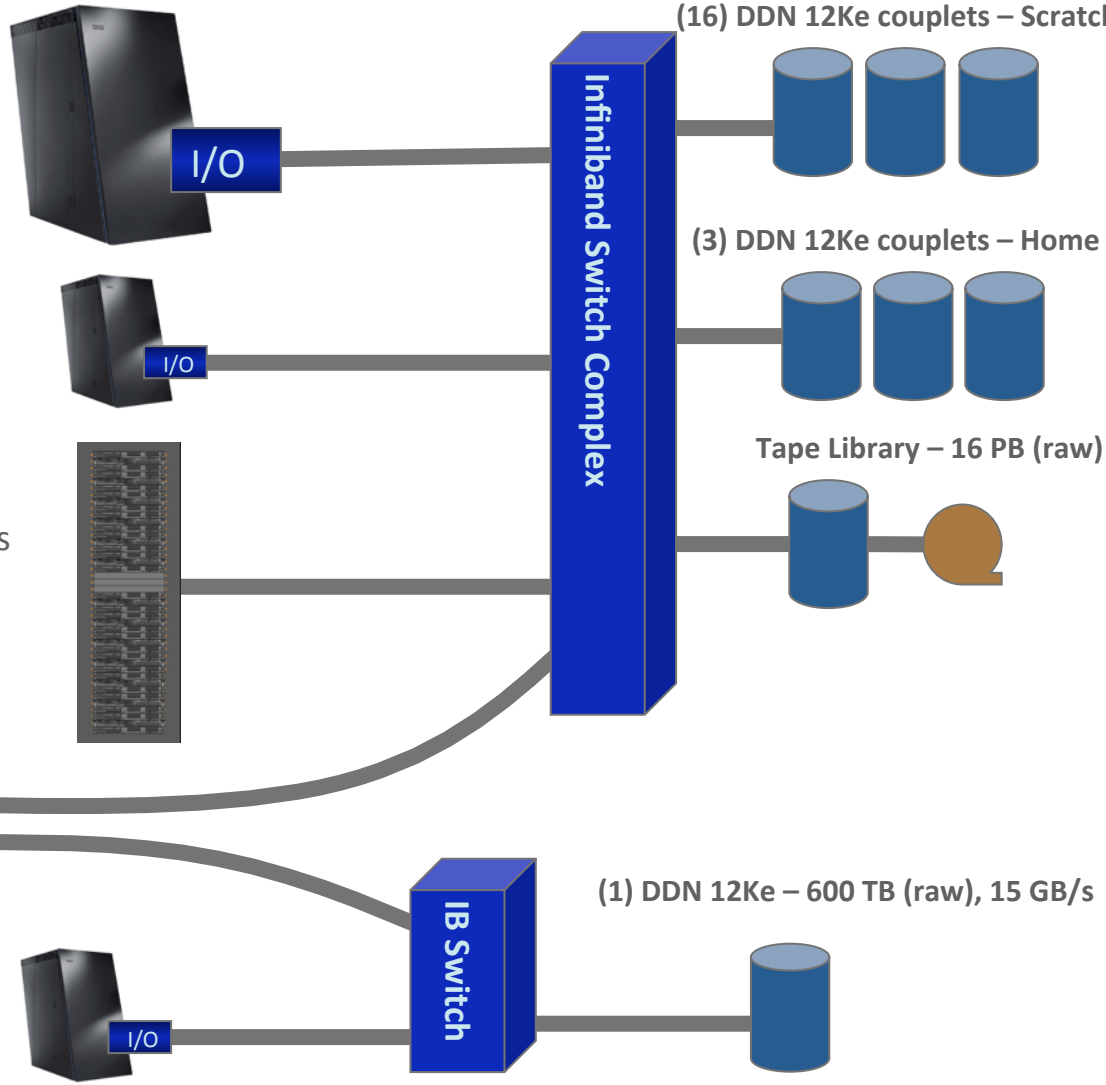
ALCF BG/Q Systems

Mira
48 racks/768K cores
10 PF

Cetus (Dev)
4 racks/64K cores
838 TF

Cooley (Viz)
126 nodes/1512 cores
126 nVIDIA Tesla K80
220 TF

Vesta (Dev)
2 racks/32K cores
419 TF

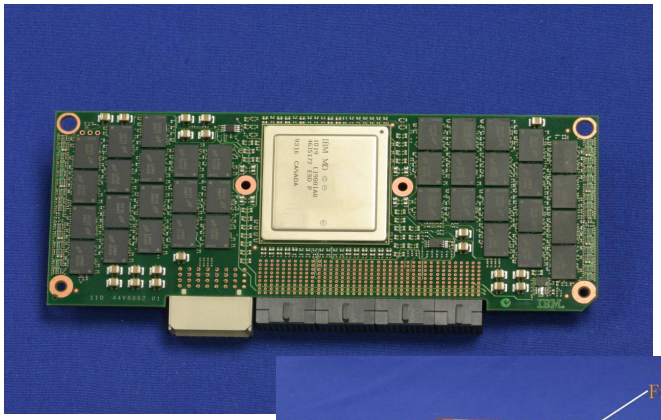


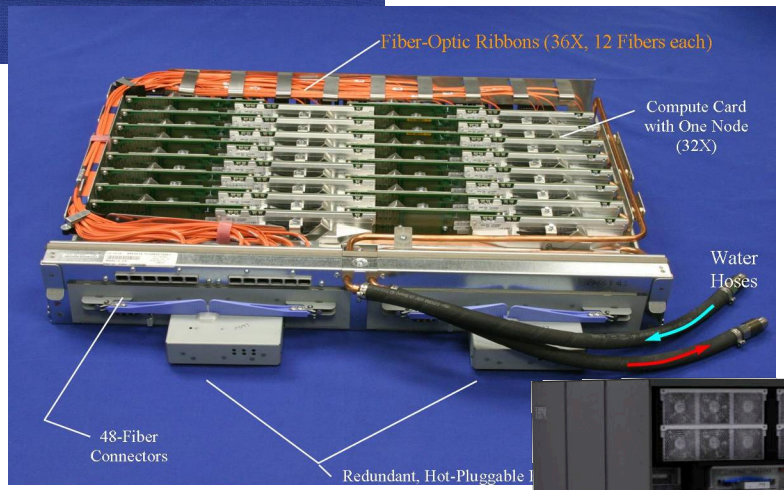
Networks – 100Gb
(via ESnet, internet2
UltraScienceNet,)

Blue Gene/Q Components

Compute Node:

- **Processor:**
  - 18 cores (205 GF)
  - Memory Controller
  - Network Interface
- **Memory:**
  - 16 GB DDR3
  - 72 SDRAMs, soldered
- **Network connectors**





Node Card Assembly or Tray

- 32 Compute Nodes (6.4 TF)
- Electrical network
- Fiber optic modules and link chips
- Water cooling lines
- Power supplies

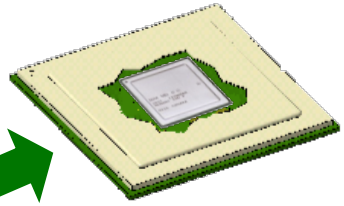
Rack

- 32 Node Trays (1024 nodes) (205 TF)
- 5D Torus Network (4x4x4x8x2)
- 8 IO nodes
- Power Supplies

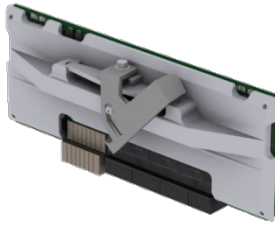
Blue Gene/Q

1. Chip:
16+2 cores

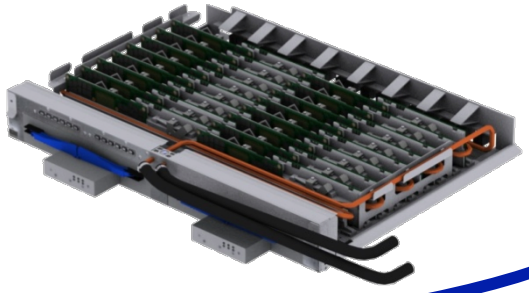
2. Single Chip Module



3. Compute card:
One chip module,
16 GB DDR3 Memory,
Heat Spreader for H_2O Cooling



4. Node Card:
32 Compute Cards,
Optical Modules, Link Chips; 5D Torus

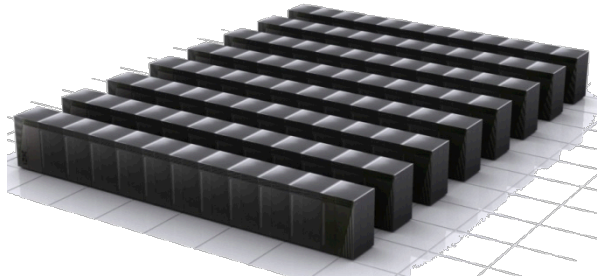


5a. Midplane:
16 Node Cards

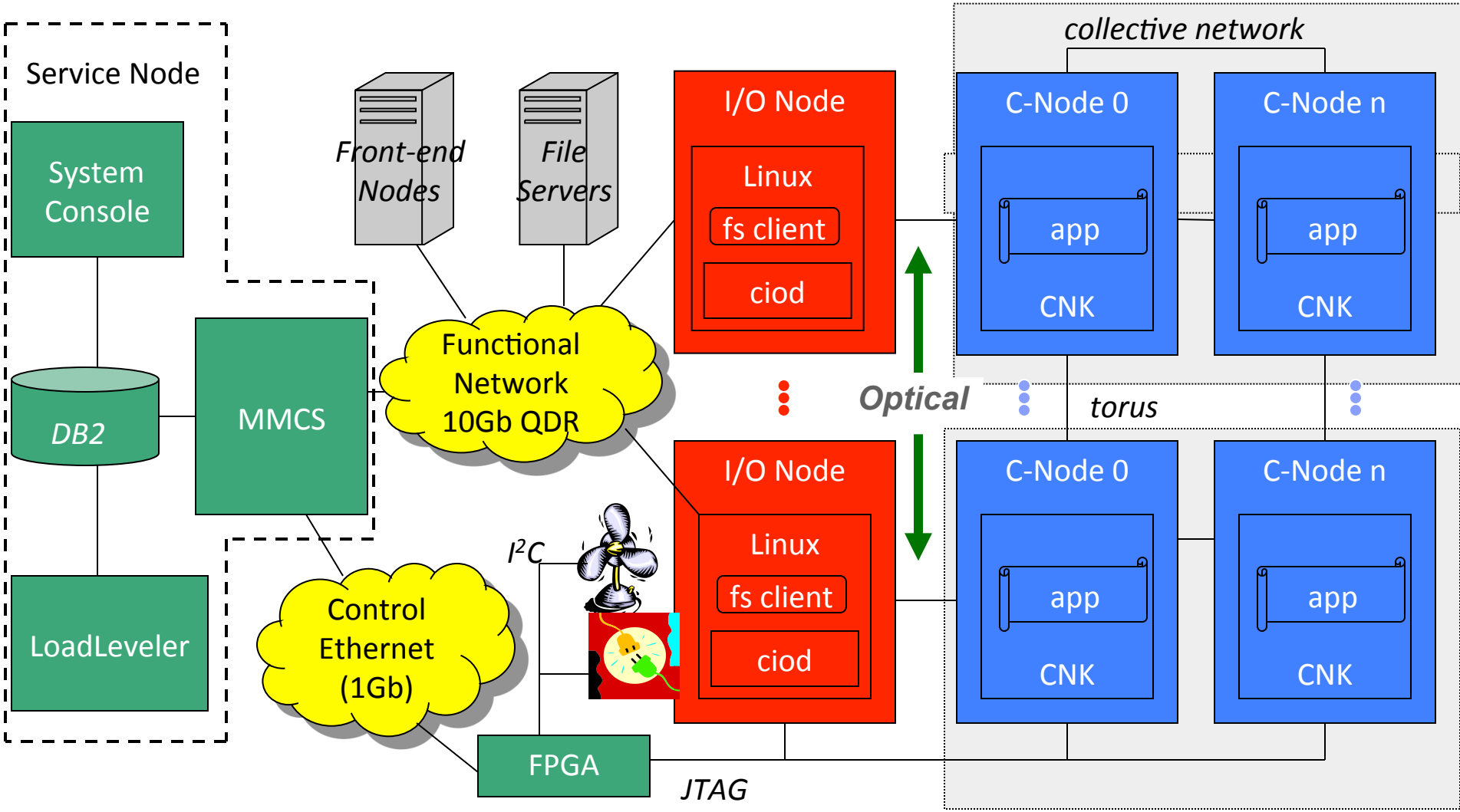
5b. IO drawer:
8 IO cards w/16 GB
8 PCIe Gen2 x8 slots
3D I/O torus

6. Rack: 2 Midplanes

7. System:
48 racks, 10PF/s

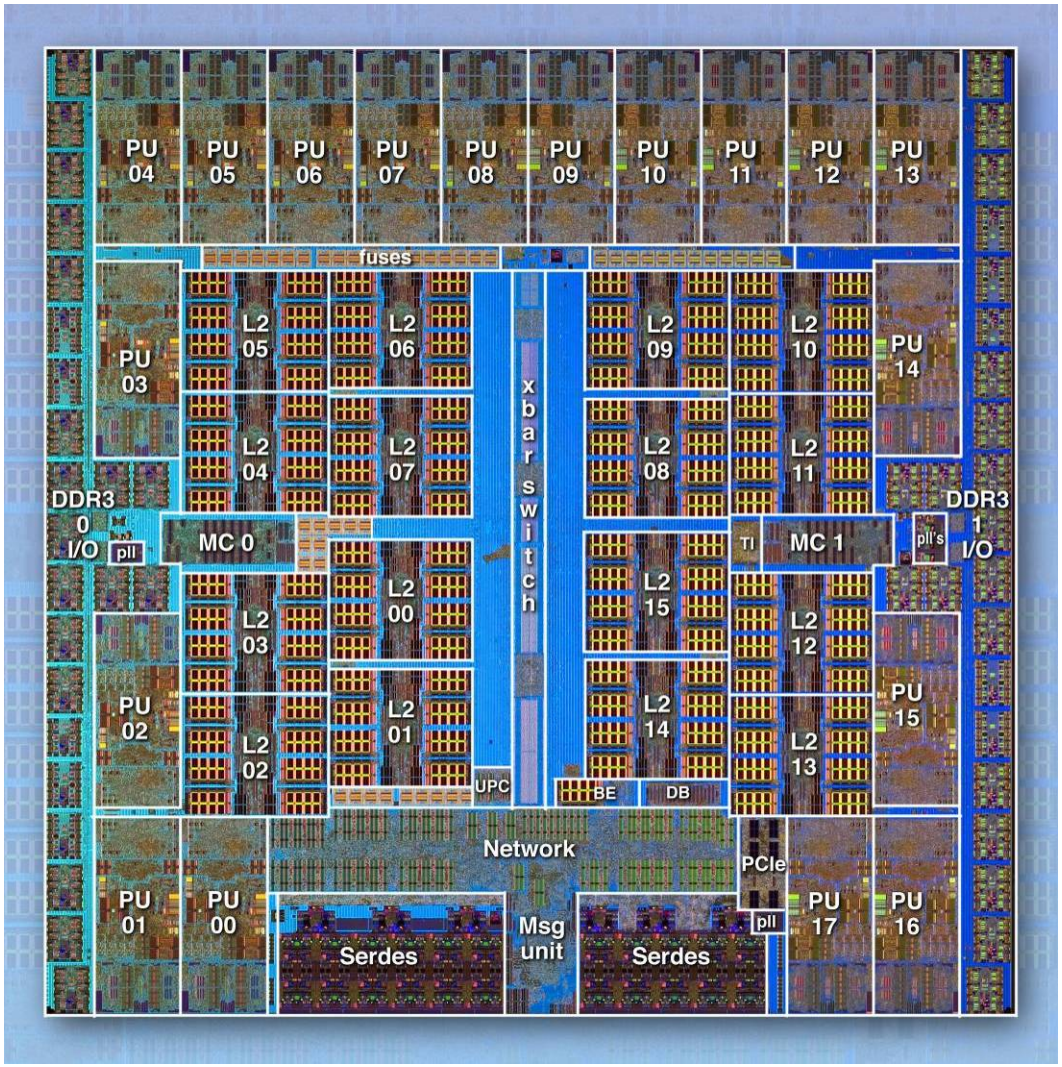


Blue Gene System Architecture



Questions?

BlueGene/Q Compute Chip



It's big!
- 360 mm² Cu-45 technology (SOI)
- ~ 1.47 B transistors

18 Cores
- 16 compute cores
- 17th core for system functions (OS, RAS)
- plus 1 redundant processor
- L1 I/D cache = 16kB/16kB

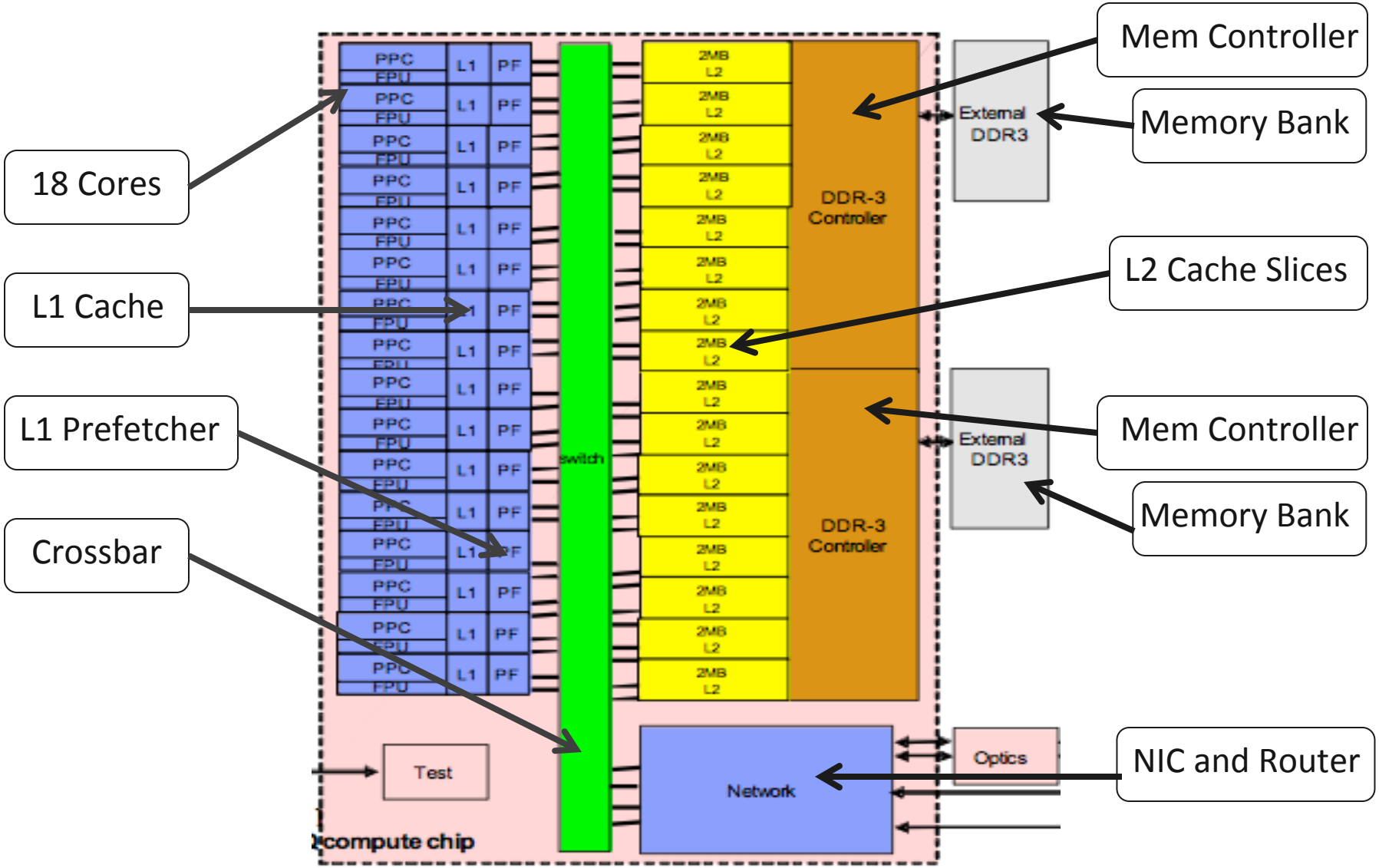
Crossbar switch
- Each core connected to shared L2
- Aggregate read rate of 409.6 GB/s

Central shared L2 cache
- 32 MB eDRAM
- 16 slices

Dual memory controller
- 16 GB external DDR3 memory
- 42.6 GB/s bandwidth

On Chip Networking
- Router logic integrated into BQC chip
- DMA, remote put/get, collective operations
- 11 network ports

BG/Q Chip, Another View

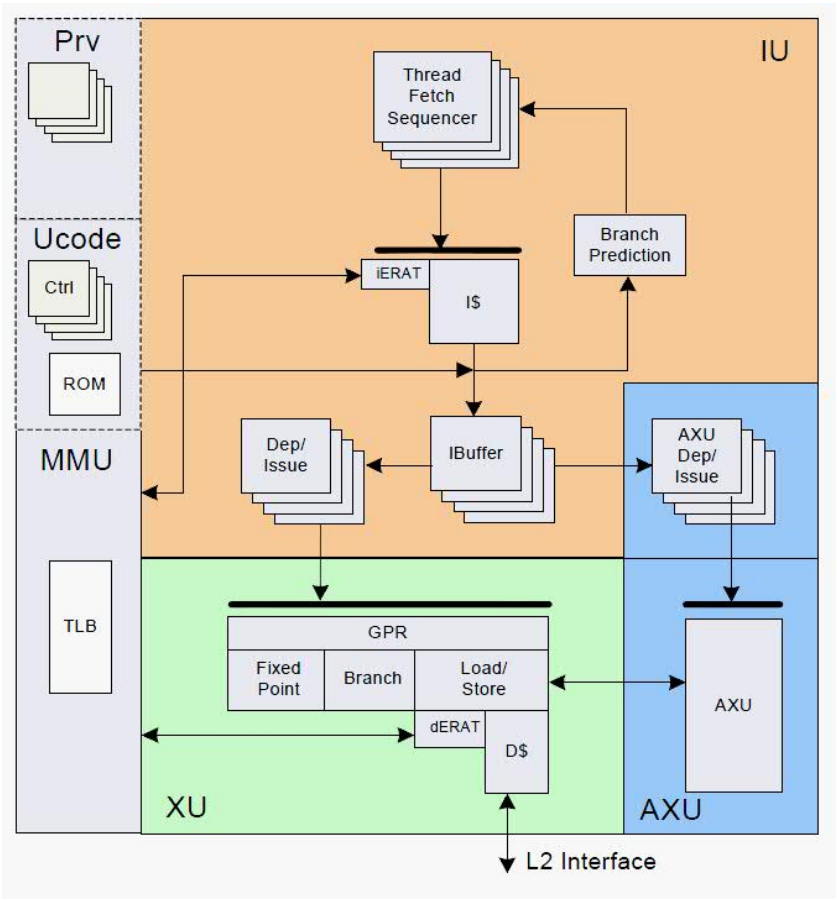


BG/Q Core

- Full PowerPC compliant 64-bit CPU, PowerISA v.206
  - *Plus QPX floating point vector instructions*
- Runs at 1.6 GHz
- In-order execution
- 4-way Simultaneous Multi-Threading
- Registers: 32 64-bit integer, 32 256-bit floating point

Functional Units:

- IU – instructions fetch and decode
- XU – Branch, Integer, Load/Store instructions
- AXU – Floating point instructions
  - Standard PowerPC instructions
  - QPX 4 wide SIMD
- MMU – memory management (TLB)

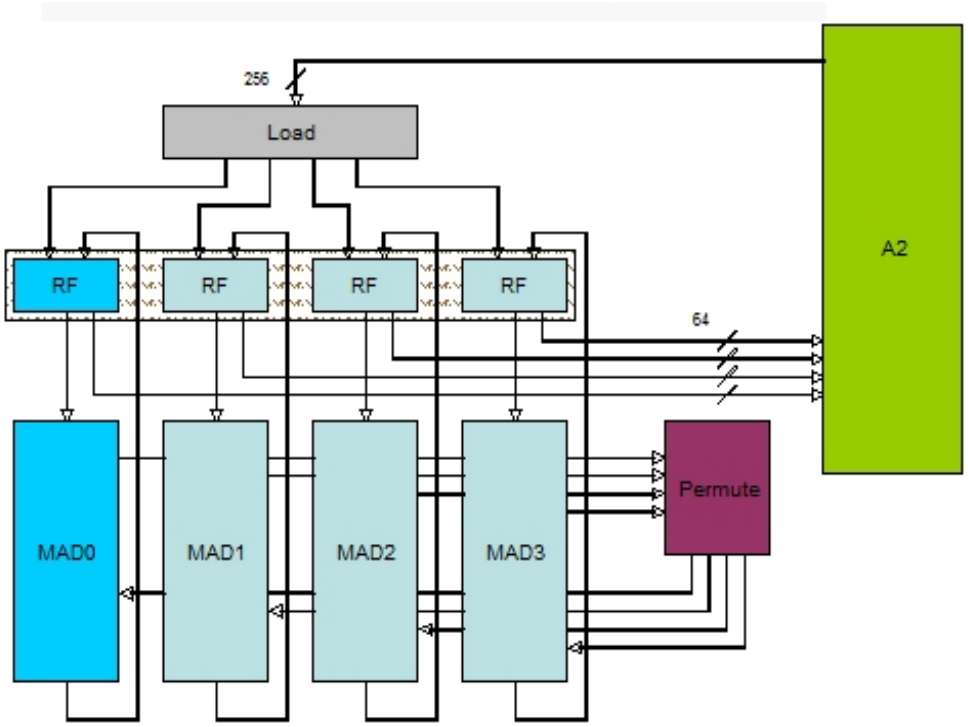


Instruction Issue:

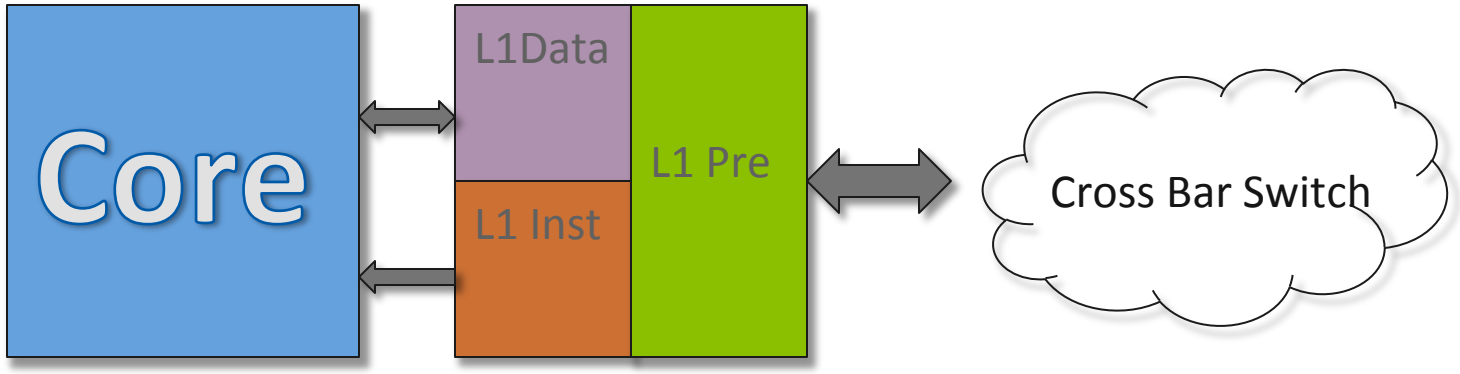
- 2-way concurrent issue 1 XU + 1 AXU
- A given thread may only issue 1 instruction per cycle
- Two threads may issue 1 instruction each cycle

QPX Overview

- Unique 4 wide double precision SIMD instructions extending standard PowerISA with:
  - Full set of arithmetic functions
  - Load/store instructions
  - Permute instructions to reorganize data
- 4 wide FMA instructions allow 8 flops/inst
- FPU operates on:
  - Standard scale PowerPC FP instructions
  - 4 wide SIMD instructions
  - 2 wide complex arithmetic SIMD arithmetic
- Standard 64 bit floating point registers are extended to 256 bits
- Attached to AXU port of A2 core
- A2 issues one instruction/cycle to AXU
- 6 stage pipeline
- Compiler can generate QPX instructions
- Intrinsic functions mapping to QPX instructions allow easy QPX programming



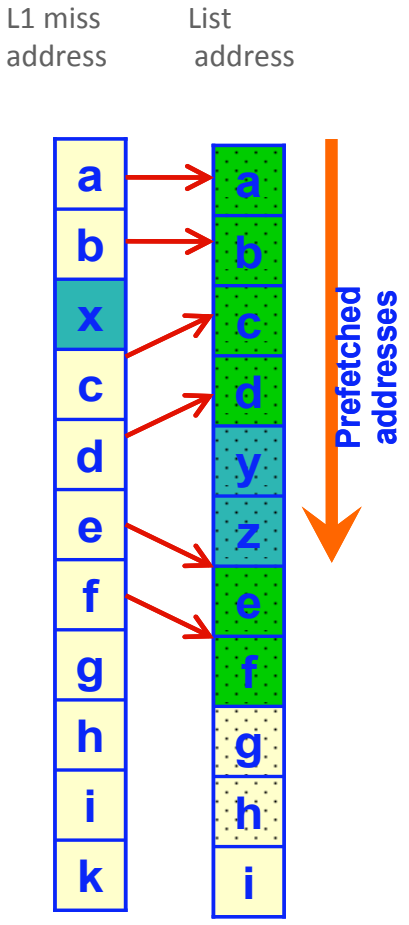
L1 Cache & Prefetcher



- **Each Core has it's own L1 cache and L1 Prefetcher**
- **L1 Cache:**
  - **Data**: 16KB, 8 way set associative, 64 byte line, 6 cycle latency
  - **Instruction**: 16KB, 4 way set associative, 3 cycle latency
- **L1 Prefetcher (L1P):**
  - 1 prefetch unit for each core
  - 32 entry prefetch buffer, entries are 128 bytes, 24 cycle latency
  - Operates in List or Stream prefetch modes
  - Operates as write-back buffer

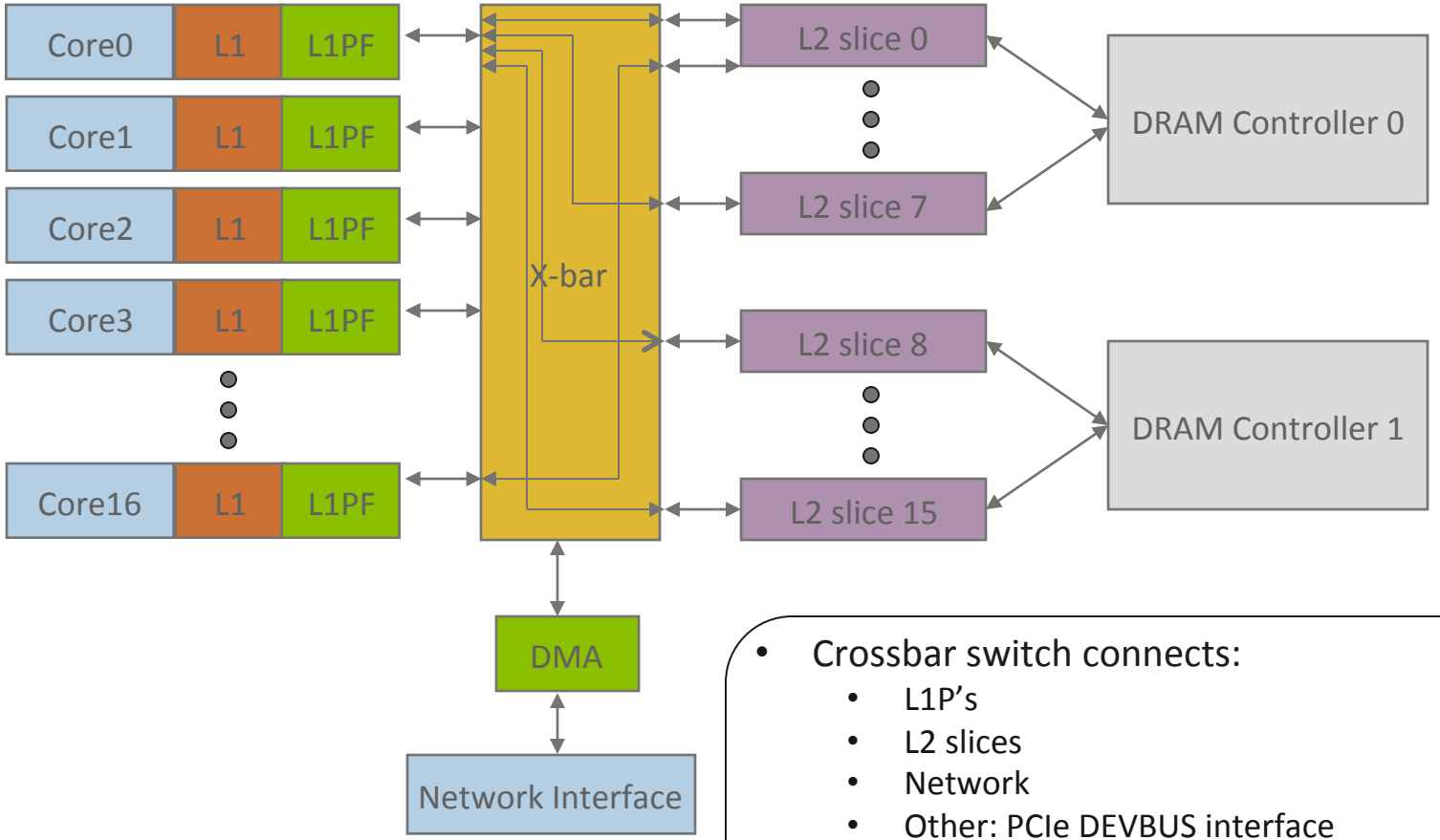
L1 Prefetcher

- **Each core has a prefetch unit that attempts to reduce the latency for L1 misses**
- **Prefetch buffer holds 32 128 byte cache lines**
- **Stream Prefetching:**
  - Default mode
  - Attempts to identify sequences of increasing contiguous loads based on L1 misses and prefetch data for upcoming loads
  - Adaptively adapts prefetch depth from 16 streams x 2 deep to 4 streams x 8 deep
- **List Prefetching:**
  - 4 units per core, 1 per hardware thread
  - Allows prefetching of arbitrary memory access patterns accessed repeatedly
  - Activated by program directives bracketing sections of code
  - Record pattern on first loop iteration and playback for subsequent iterations
  - List is adaptively adjusted for missing or extra cache misses



List-based “perfect” prefetching has tolerance for missing or extra cache misses

BG/Q Crossbar Switch



- Crossbar switch connects:
  - L1P's
  - L2 slices
  - Network
  - Other: PCIe DEVBUS interface
- Aggregate bandwidth across slices:
  - Read: 409.6 GB/s
  - Write: 204.8 GB/s

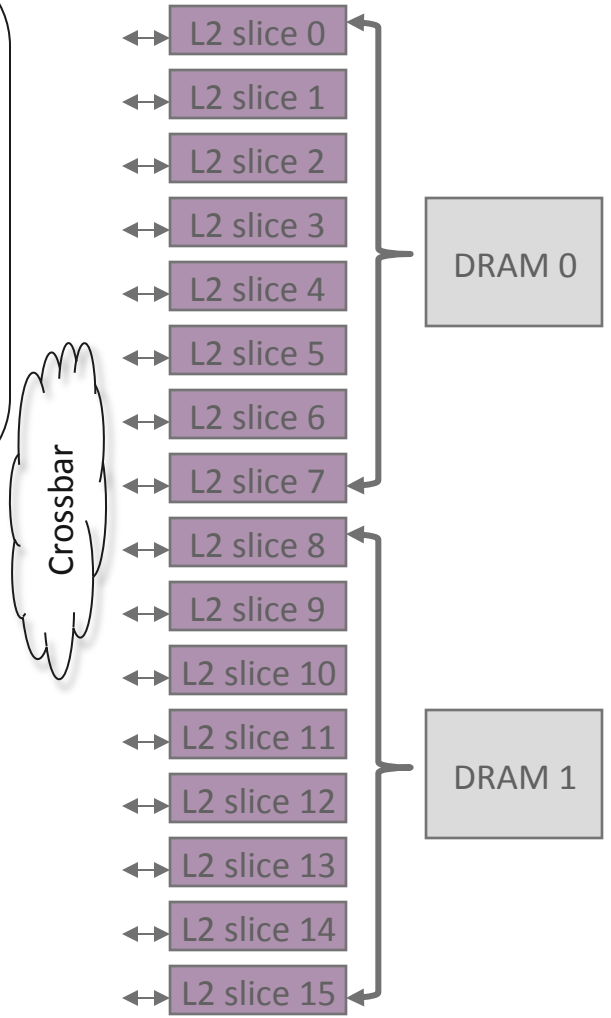
L2 Cache & Memory

L2 Cache:

- Shared by all cores
- Serves a point of coherency, generates L1 invalidations
- Divided into 16 slices connected via crossbar switch to each core
- 32 MB total, 2 MB per slice
- 16 way set assoc., write-back, LRU replacement, 82 cycle latency
- Supports memory speculation and atomic memory operations
- Has prefetch capabilities based on hints from L1P

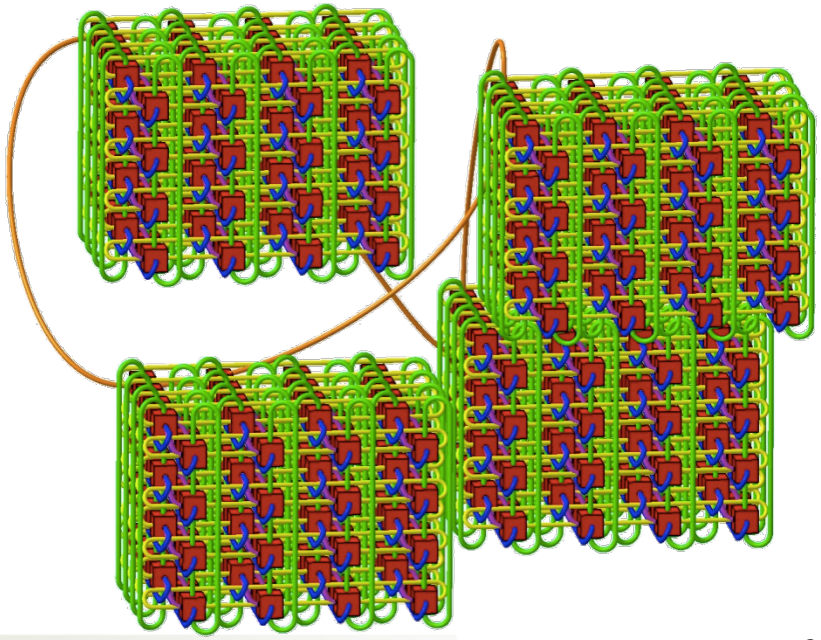
Memory:

- Two on-chip memory controllers
- Each connects to 8 L2 slices via 2 ring buses
- Each controller drives a 16+2 byte DDR-3 channel at 1.33 Gb/s
- Peak bandwidth is 42.67 BG/s (excluding ECC)
- Latency > 350 cycles



Inter-Processor Communication

- **5D torus network:**
  - Achieves high nearest neighbor bandwidth while increasing bisectional bandwidth and reducing hops vs 3D torus
  - Allows machine to be partitioned into independent sub machines
    - No impact from concurrently running codes.
  - Hardware assists for collective & barrier functions over COMM_WORLD and rectangular sub communicators
  - Half rack (midplane) is 4x4x4x4x2 torus (last dim always 2)
- **No separate Collectives or Barrier network:**
  - Single network used for point-to-point, collectives, and barrier operations
- **Additional 11th link to IO nodes**
- **Two type of network links**
  - Optical links between midplanes
  - Electrical inside midplane



Network Performance

- **Nodes have 10 links with 2 GB/s raw bandwidth each**
  - Bi-directional: send + receive gives 4 GB/s
  - 90% of bandwidth (1.8 GB/s) available to user
- **Hardware latency**
  - ~40 ns per hop through network logic
  - Nearest: 80ns
  - Farthest: 3us (96-rack 20PF system, 31 hops)
- **Network Performance**
  - Nearest-neighbor: 98% of peak
  - Bisection: > 93% of peak
  - All-to-all: 97% of peak
  - Collective: FP reductions at 94.6% of peak
  - Allreduce hardware latency on 96k nodes ~ 6.5 *u*s
  - Barrier hardware latency on 96k nodes ~ 6.3 *u*s

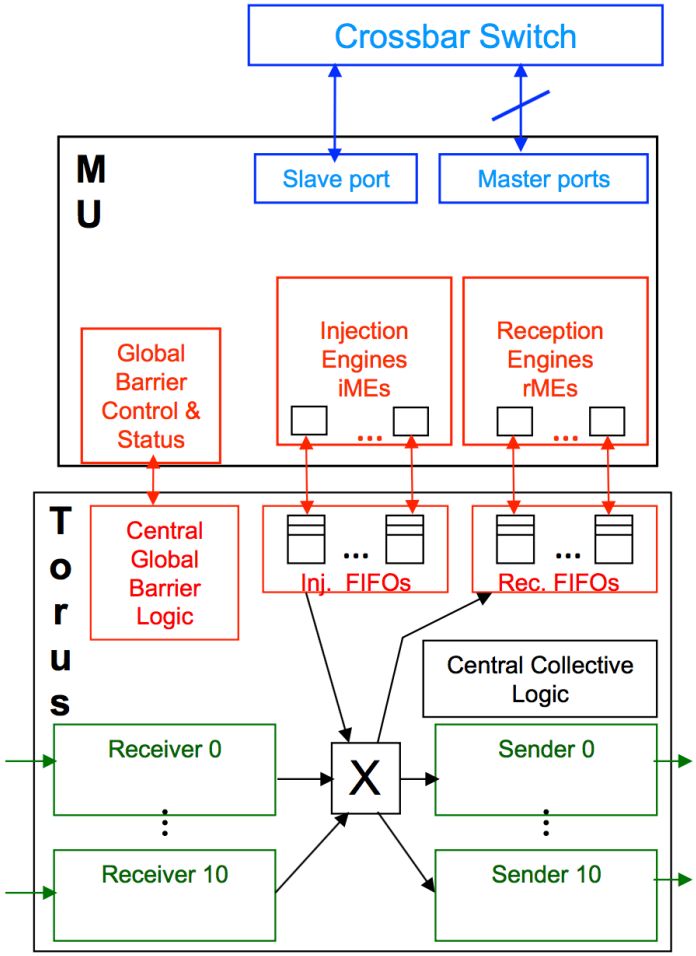
Network Interface and Router

Network Unit (Torus Router)

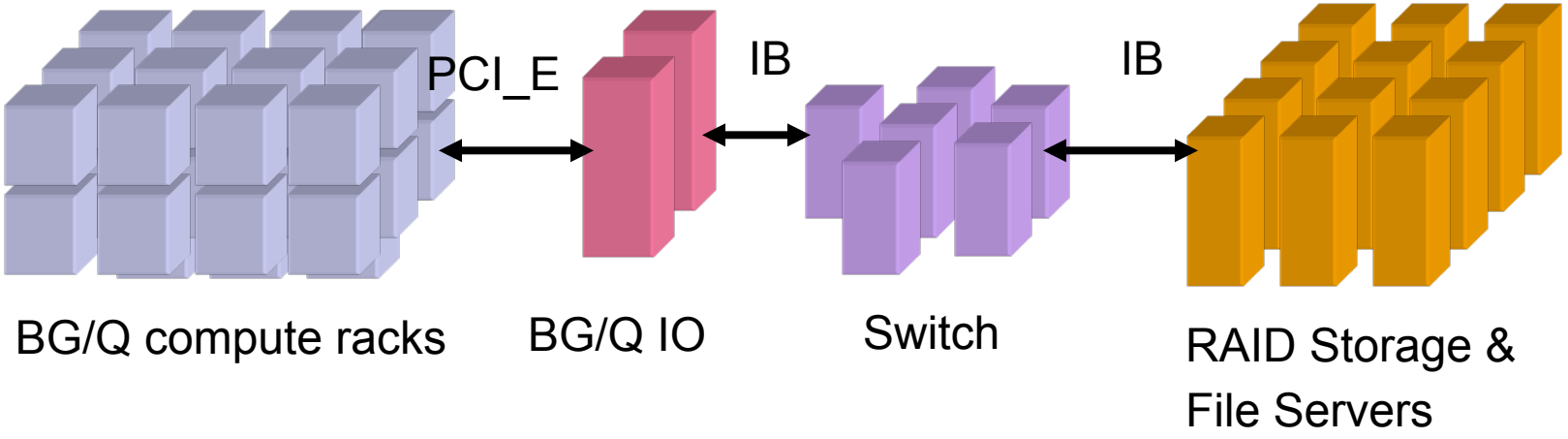
- Each chip has 11 network send units and 11 receive units:
  - Each can transmit and receive at 2 GB/s simultaneously
  - 10 Torus links, 1 IO link, total bandwidth of 44 GB/S
- 16 hardware network injection & reception FIFOs
  - Packets placed in injection FIFOs are sent out via the Sender
  - Packets received for the node are placed in the reception FIFOs
- Packets from Receivers passing through can go directly to Senders
- Receivers contain 7 Virtual Channel packet buffers: point-to-point, high priority, system, collectives
- Collective operations are handled by Central Collective Logic

Messaging Unit (Network Interface)

- Interface between the network and the BG/Q memory system
  - Injects and pulls packets from network FIFOs
  - Supports direct puts, remote gets, and memory FIFO messages
- Maintains pointers to FIFOs in main memory
  - 544 injection memory FIFOs, 272 memory reception FIFOs
  - Messages sent by writing descriptor into injection FIFO
- 16 Injection Message Engines and 16 Reception Message Engines
  - Injection engines are assigned a descriptor, pull data and packetize
  - Reception engines pull data from reception FIFOs and write to in-memory FIFOs, or specified memory location



BG/Q IO



IO is sent from Compute Nodes to IO Nodes to storage network

- IO Nodes handle function shipped IO calls to parallel file system client
- IO node hardware is functionally identical to compute node hardware
- IO nodes run Linux and mount file system
- "Bridge" Compute Nodes use 1 of the 11 network links to link to IO nodes
- Each IO node connects to 2 bridge nodes (one in each of two compute partitions)
  - Pairs of IO nodes are shared by pairs of compute nodes
  - Only at smallest HW partition size, no sharing between larger partitions.

Blue Gene/Q Software High-Level Goals & Philosophy

- Facilitate extreme scalability
  - Extremely low noise on compute nodes
- High reliability: a corollary of scalability
- Familiar programming modes such as MPI and OpenMP
- Standards-based when possible
- Open source where possible
- Facilitate high performance for unique hardware:
  - Quad FPU, DMA unit, List-based prefetcher
  - TM (Transactional Memory), SE (Speculative Execution)
  - Wakeup-Unit, Scalable Atomic Operations
- Optimize MPI and native messaging performance
- Optimize libraries
- Facilitate new programming models

Blue Gene/Q Software

- Standards-based programming environment
  - Linux development environment: familiar GNU toolchain with glibc, pthreads
  - XL Compilers C, C++, Fortran with OpenMP 3.1
  - Debuggers: Totalview
  - Tools: HPC Toolkit, TAU, PAPI, Valgrind
- Message Passing
  - Scalable MPICH2 providing MPI 2.2 with extreme message rate
  - Efficient intermediate (PAMI) and low-level (SPI) message libraries
    - documented and open source
  - PAMI layer allows easy porting of runtimes like GA/ARMCI, Berkeley UPC, etc
- Compute Node Kernel (CNK) eliminates OS noise
  - File I/O offloaded to I/O nodes running full Red Hat Linux
  - GLIBC environment with a few restrictions for scaling
- Flexible and fast job control – with high availability
  - Noise-free partitioned networks
  - Integrated HPC, HTC, MPMD, and sub-block jobs

BG/Q Special Features

- 4-wide SIMD floating point unit (QPX)
- Transactional Memory & Speculative Execution
- Fast memory based atomic operations
- Stream and list based prefetching
- WakeUp Unit
- Universal Performance Counters

Fast Atomics

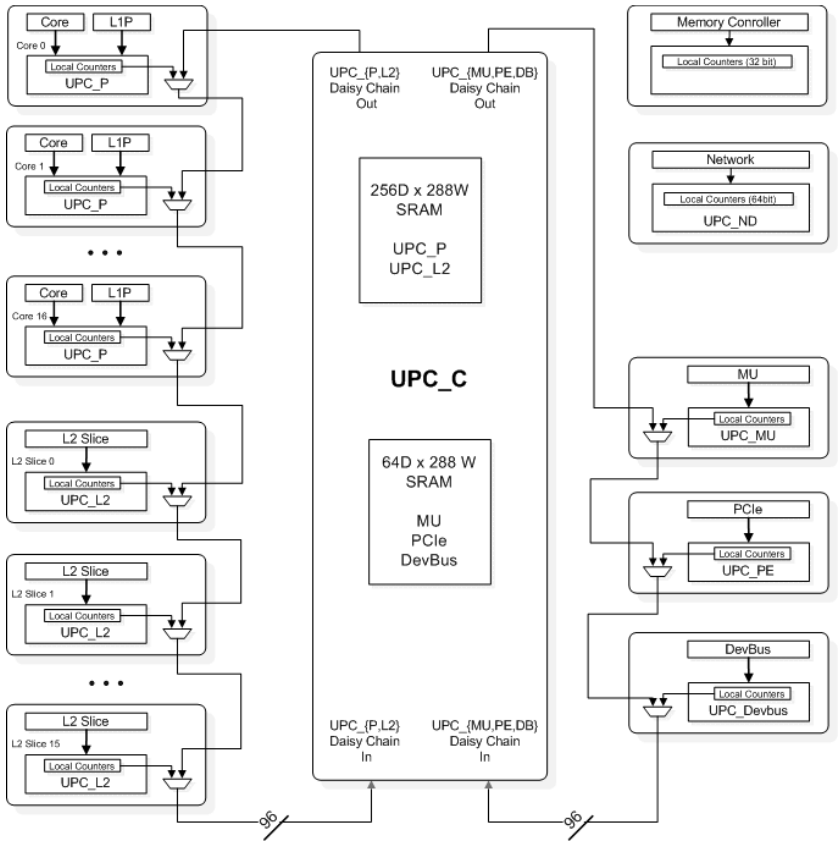
- **Provided in hardware by the L2**
- **8 byte load & store operations that can alter the value at any memory address**
- **Atomics use standard load & store instructions with special high order address bits**
- **Allow fast synchronization and concurrent data structures – a ticket lock can by implemented to run 30x faster**
- **Load Operations:**
  - LoadClear, LoadIncrement, LoadDecrement, LoadIncrementBounded, LoadDecrementBounded
- **Store Operations:**
  - StoreAdd, StoreAddCoherenceOnZero, StoreTwin, StoreOr, StoreXor, StoreMaxUnsigned, StoreMaxSigned
- **Memory for Atomics must be reserved with Kernel_L2AtomicsAllocate( )**

WakeUp Unit

- Each core includes a WakeUp Unit
- Improves overall performance by reducing the cost of spin or polling loops
  - Polling threads issue instructions that occupy issues slots
- Threads can configure the WakeUp unit to watch for writes to a range of memory addresses
- Threads can be suspended until a watched address is written to
- Thread is reactivated when watched address is written to
- Improves power efficiency and resource utilization

Hardware Performance Counters

- Universal Performance Counter (UPC) unit collects hardware performance events from counters on:
  - 17 cores
  - L1P's
  - Wakeup Units
  - 16 L2 slices
  - Message, PCIe, and DEVBUS units
- Wide range of hardware events
- Network Unit maintains a separate set of counters
- Accessible through BGPM API



Transactional Memory and Speculative Execution

- **Transactional Memory implemented in L2:**
  - Sections of code are annotated to be executed atomically and in isolation using pragma *tm_atomic*
  - Changes from speculative threads kept separate from main memory state
  - Speculatively written data only available to thread writing it
  - At end of speculative section can revert or commit changes
  - Hardware identifies conflicts: *read-after-write, write-after-read, write-after-write*
  - Can store up to 30MB of speculative state
- **Speculative Execution implemented in L2:**
  - Sections of code are annotated to be executed speculatively in parallel using pragmas*: speculative for, speculative sections*
  - Sequential code is partitioned into tasks which are executed speculatively in parallel
  - Data written by sequentially earlier threads is forwarded to later threads
  - Conflicts are detected by hardware at 8 bytes resolution

Questions?